

RECEIVED  
CENTRAL FAX CENTER

APR 30 2007

## WILLIAMS, MORGAN &amp; AMERSON, P.C.

7676 Hillmont, Suite 250, Houston, TX 77040  
(713) 934-7000 Fax (713) 934-7011**Fax: Art Unit 2195**

To:	USPTO	From	Jeffrey A. Pyle
Fax:	571 273 8300	Phone:	(713) 934-4053
No. of Pages:	18	Date:	April 30, 2007
Re:	10/044,707 Appeal Brief	File:	2000.052200
<input type="checkbox"/> Urgent <input type="checkbox"/> For Review <input type="checkbox"/> For Your File <input type="checkbox"/> Please Reply <input type="checkbox"/> Please Handle			

\* w/o coversheet

ORIGINAL: \_\_\_ Will follow ☒ Will not follow

## • Comments:

Confirmation No. 1281

## CONFIDENTIALITY NOTE

The documents accompanying this facsimile transmission contain information from the law firm of Williams, Morgan & Amerson which may be confidential and/or privileged. The information is intended to be for the use of the individual or entity named on this transmission sheet. If you are not the intended recipient, be aware that any disclosure, copying, distribution or use of the contents of this faxed

RECEIVED  
CENTRAL FAX CENTER

## IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APR 30 2007

In Re Application Of:	Dale E. Gulick	Examiner:	J. To
Serial No.:	10/044,707	Group Art Unit:	2195
Filed:	January 11, 2002	Att'y Docket:	2000.052200
For: Processing Tasks With Failure Recovery		Client No.:	TT4036
		Confirmation No.:	1281

APPEAL BRIEF

Commissioner for Patents  
PO Box 1450  
Alexandria, VA 22313-1450

Sir:

Applicants hereby submit this Appeal Brief to the Board of Patent Appeals and Interferences in response to the final Office Action dated November 28, 2006. A Notice of Appeal was filed on February 28, 2007.

The fee for filing this Appeal Brief is \$500, and the Commissioner is authorized to deduct said fees from Williams, Morgan & Amerson Deposit Account No. 50-0786/2000.052200/PYL.

**I. REAL PARTY IN INTEREST**

The present application is owned by Advanced Micro Devices, Inc.

**II. RELATED APPEALS AND INTERFERENCES**

There are no related appeals or interferences of which Applicants, Applicants' legal representative, or the Assignee is aware that will directly affect or be directly affected by or have a bearing on the decision in this appeal.

**III. STATUS OF THE CLAIMS**

Claims 1-4, 6-11, 13-18 and 20-28 are pending in the application. Claims 1-29 were originally filed. The "final" Office Action rejected claims 1-4, 6-11, 13-18, 20 and 21. Claims 22-28 were allowed. More particularly, the final Office Action rejected:

APR 30 2007

- claims 8-11 and 13-14 under 35 U.S.C. §101 as directed to non-statutory subject matter;
- claims 8-11 and 13-14 as indefinite under 35 U.S.C. §112, ¶2;
- claims 1, 4, 6-11, 13-15, 18, and 20-21 as obvious under 35 U.S.C. §103(a) over U.S. Letters Patent 6,148,322 ("Sand *et al.*"); and
- claims 2-3 and 16-17 as obvious under 35 U.S.C. §103(a) over Sand *et al.* combined with U.S. Letters Patent 5,012,409 ("Fletcher").

Applicant appeals from each of the rejections.

#### IV. STATUS OF AMENDMENTS

An amendment was submitted after the final Office Action. The Advisory Action indicated that the amendment was not entered.

#### V. SUMMARY OF CLAIMED SUBJECT MATTER

Computer systems today process a fairly large number of tasks at any given time. Typically, a computing system may be executing several threads at one time, each of which comprised of a number of "tasks". As the number of tasks that are processed increases, the likelihood that some of these tasks may not successful complete (because of errors, for example) also increases. Errant or hung tasks, for example, may adversely affect the performance of the computer system. As such, recovery from these failed tasks is desirable.

The present invention is best shown in FIG. 8 and FIG. 9 of the application, both of which are reproduced below, and is discussed in the specification at p. 26, line 9 – p. 36, line 2. One aspect of the present invention includes a method for processing tasks with failure recovery. The method (e.g., 805, FIG. 9) includes storing one or more tasks (815(1)-815(n), FIG. 8) in a queue (e.g., 810, FIG. 8), wherein each task has an associated exit routine (p. 31, lines 1-13), and determining at least one task to process based on a priority scheme (e.g., at 930, FIG. 9; p. 26, line 21-p.22, line 7). The method further includes processing (e.g., p. 30, lines 1-8) the at least one task (e.g., at 960, FIG. 9), and calling the exit routine (p. 31, lines 1-13) based on determining that the task has not completed processing within a preselected period of time (p. 27, lines 18-p. 28, line 10).

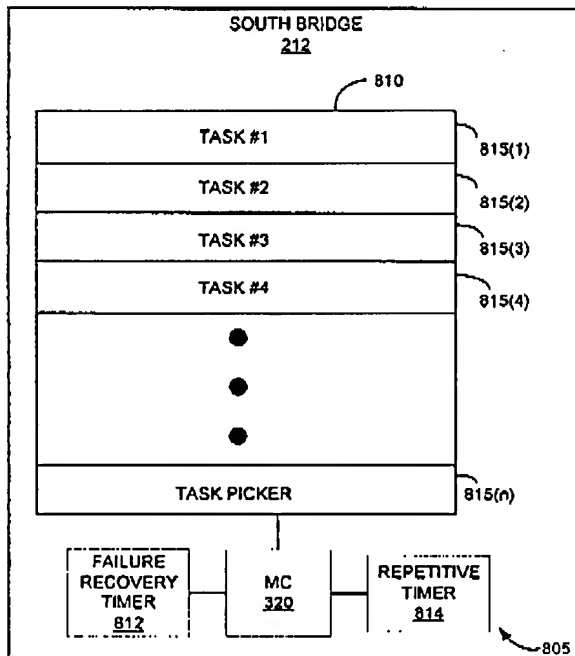


Fig. 8

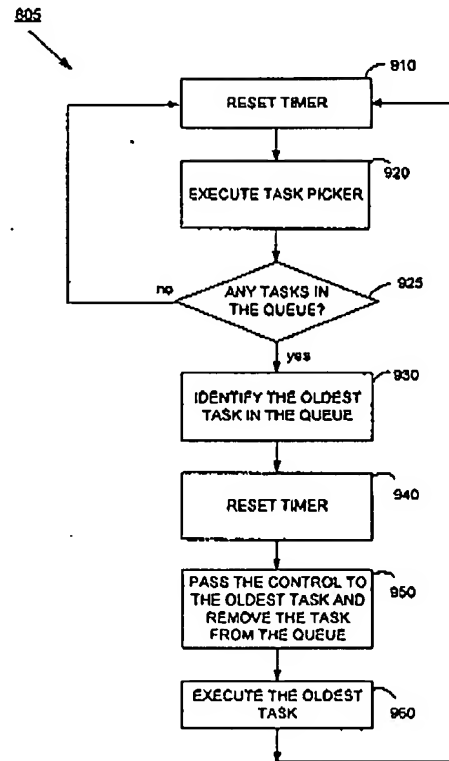


Fig. 9

In another aspect of the invention, an apparatus is provided for processing tasks with failure recovery. The apparatus comprises a queue (e.g., 810, FIG. 8) and a controller (e.g., 320, FIG. 8). The queue includes a task picker (e.g., 815(n), FIG. 8) stored therein. The controller (e.g., 320, FIG. 8), which is communicatively coupled to the queue, is adapted to determine if at least one task (e.g., 815(1)-815(n-1), FIG. 8) other than the task picker is stored in the queue and execute the task based on determining that at least one task other than the task picker is stored in the queue. The controller is further adapted to execute the task picker in response to executing the task and to continue to execute the task picker until a preselected event occurs (p. 27, lines 18-p. 28, line 10).

Turning now to the language of the claims, with respect to claim 1, a computer implemented method, the invention comprises:

- storing one or more tasks (815(1)-815(n-1), FIG. 8) in a queue (e.g., 810, FIG. 8), wherein each task has an associated exit routine (p. 31, lines 1-13);
- determining at least one task to process based on a priority scheme (e.g., at 930, FIG. 9; p. 26, line 21-p.22, line 7);
- processing the at least one task (e.g., p. 30, lines 1-8); and
- calling the exit routine based on determining that the task has not completed processing within a preselected period of time (p. 27, lines 18-p. 28, line 10).

With respect to claim 8, a computing apparatus, the invention comprises:

- a queue (e.g., 810, FIG. 8) having a task picker (815(n), FIG. 8) stored therein, the task picker being configured to:
  - determine if at least one task (815(1)-815(n-1), FIG. 8) other than the task picker is stored in the queue;
  - transfer control to the at least one task other than the task picker based on determining that the at least one task other than the task picker is stored in the queue so that the at least one task other than the task picker can execute; and
  - execute (e.g., p. 30, lines 1-8) in response to the at least one task other than the task picker completing execution and continue executing until a preselected event occurs (p. 27, lines 18-p. 28, line 10).

With respect to claim 15, an article comprising one or more machine-readable storage media containing instructions that when executed enable a processor to:

- store one or more tasks (815(1)-815(n), FIG. 8) in a storage space (e.g., 810, FIG. 8), wherein each task has an associated exit routine (p. 31, lines 1-13);
- determine at least one task to process based on a priority scheme (e.g., at 930, FIG. 9; p. 26, line 21-p.22, line 7);
- process (e.g., p. 30, lines 1-8) the at least one task; and
- call the exit routine based on determining that the task cannot be processed to completion.

With respect to claim 22, an apparatus, comprising:

a queue (e.g., 810, FIG. 8) having a task picker (815(n), FIG. 8) stored therein, the queue adapted to store one or more tasks (815(1)-815(n), FIG. 8), and the task picker being configured to:

- select a task from the queue to execute based on a priority scheme (e.g., at 930, FIG. 9; p. 26, line 21-p.22, line 7);
- transfer control to the task so that the task can execute (e.g., p. 30, lines 1-8);
- a failure recovery timer (e.g., 812, FIG. 8) to generate an interrupt at preselected time intervals, wherein each preselected time interval is greater than the time it takes for each of the tasks stored in the queue to execute; and
- a controller (e.g. 330, FIG. 8) adapted to:
  - determine if the task completes execution within the preselected time interval;
  - terminate the task in response to determining that the task failed to complete within the preselected time interval (p. 27, lines 18-p. 28, line 10); and
  - execute the task picker in response to terminating the task.

Note that the references set forth above are to the disclosed embodiment(s) for illustrative purposes as required by rule and do not limitation the claims.

## VI. GROUND OF REJECTION TO BE REVIEWED ON APPEAL

- A. Whether claims 8-11 and 13-14 under 35 U.S.C. §101 are directed to non-statutory subject matter.
- B. Whether claims 8-11 and 13-14 are indefinite under 35 U.S.C. §112, ¶2.
- C. Whether claims 1, 4, 6-11, 13-15, 18, and 20-21 are obvious under 35 U.S.C. §103(a) over U.S. Letters Patent 6,148,322 ("Sand et al.").
- D. Whether claims 2-3 and 16-17 are obvious under 35 U.S.C. §103(a) over Sand *et al.* combined with U.S. Letters Patent 5,012,409 ("Fletcher").

## VII. ARGUMENT

**A. CLAIMS 8-11 AND 13-14 ARE DIRECTED TO  
STATUTORY SUBJECT MATTER**

The Office rejected claims 8-11 and 13-14 under 35 U.S.C. §101 as directed to non-statutory subject matter. The Office's position appears to rest on three suppositions:

- the claims are directed software without reciting hardware;
- there is no specific and asserted utility; and
- there is no established utility.

Applicant respectfully submits that each of these suppositions is wrong.

The Office states that software *per se* is non-statutory. This is an incorrect statement of the law. There is no *per se* ban on patenting software under United States jurisprudence. Notably, the Office has failed to produce any authority for this proposition. Thus, the fact that the claims "...[appear] to be comprised of software alone without claiming associated computer hardware required for execution..." is of no consequence in the analysis under 35 U.S.C. §101. The legal authority cited by the Office in support of this rejection, *i.e.*, *In re Lowry*, 32 U.S.P.Q.2d (BNA) 1031 (Fed. Cir. 1994); *In re Warmerdam*, 31 U.S.P.Q.2d (BNA) 1754 (Fed.Cir. 1994), do not stand for this proposition.

However, Applicant respectfully submits that the Office's observation that the claims are directed solely to software is incorrect. The preambles clearly recite that the claims are for a "computing apparatus." It is well established the recitations in the preamble of a claim can limit its scope and the test is whether they "breathe life and meaning" into the claims. *In re Higbee*, 527 F.2d 1405, 1407 (C.C.P.A. 1976); *Kropa v. Robie*, 187 F.2d 150, 151-52 (C.C.P.A. 1951). Applicants respectfully submit that the recitation of a "computing apparatus" necessarily "breathes life and meaning" into the claims if the claims are rendered non-statutory by its omission. Thus, the claims are not directed to software, but rather to a programmed computing apparatus.

Nevertheless, regardless of whether the claims are directed software or hardware, there is a specific and asserted utility and a well established utility. The "Summary of the Invention" clearly states that the invention is in one aspect a method and in a second aspect an apparatus for failure recovery. (p. 8, lines 1-15) The specification also notes that:

...modern computer systems are becoming more and more robust than their predecessors. Computer systems today process a fairly large number of tasks at any given time. As the number of tasks that are processed increases, the likelihood that some of these tasks may not successful complete (because of errors, for example) also increases. Errant or hung tasks, for example, may adversely affect the performance of the computer system. As such, recovery from these failed tasks is desirable.

Thus, Applicant has clearly asserted a specific utility—*i.e.*, failure recovery in the event of a failed task within a computing environment.

The Office also posits that failure recovery is not “well-established.” However, the Office’s actions in this examination refute this position, since it has managed to cite seven references it claims are on point. Still further, each of these references cites several references as well. The primary reference in this case, one that the Office alleges teaches essentially everything Applicant is trying to claim, happens to cite 18 references. There appears to be no shortage of art that the Office feels is relevant to failure recovery. The evidence of record produced by the Office therefore eviscerates the Office’s position regarding the asserted utility. Should the Office maintain its position on this point, Applicant requests clarification resolving the discrepancy between the evidence of record and the Office’s position contrary to that art.

Applicant notes that the Office has historically, for quite some time now, allowed claims such as claims 8-11 and 13-14. This impliedly means that Office policy for a number of years now has been that such claims are directed to statutory subject matter. A reversal of this policy, particularly as extreme as is presented in this case, without a concomitant change in law by Congress or the federal courts, constitutes an arbitrary and capricious act. Accordingly, such an act is reversible as a matter of law under even the highest standards of review in the federal courts.

The Office’s rejection of claims 8-11 and 13-14 under 35 U.S.C. §101 as directed to non-statutory subject matter is therefore contrary to both law and the evidence of record. Applicant respectfully submits that attempts at the clarification requested above will highlight the error in the Office’s position. Accordingly, Applicant requests that the rejections be REVERSED.

**B. CLAIMS 8-11 AND 13-14 ARE DEFINITE**



The Office also rejected claims 8-11 and 13-14 as indefinite under 35 U.S.C. §112, ¶2. The “final” Office Action states that “[t]he claim language in the following claims is not clearly understood”. (p.3, ¶ 7.a.) Applicant notes that whether the Office understands the claim language is irrelevant to the applicable legal analysis. All that is necessary is that the language be sufficiently definite that those skilled in the art can ascertain its scope. *Orthokinetics Inc. v. Safety Travel Chairs Inc.*, 1 U.S.P.Q.2d (BNA) 1081, 1088 (Fed. Cir. 1986); *Shatterproof Glass Corp. v. Libbey Owens Ford Co.*, 225 U.S.P.Q. (BNA) 634, 641 (C.C.P.A. 1985). However, the Office has failed even to allege the necessary factual predicate—that those skilled in the art would be unable to ascertain the scope of the language—much less provide any reasoning as to why that is the case. Thus, the Office has yet to establish *prima facie* that claims 8-11 and 13-14 are indefinite. Wherefore, Applicant requests that the rejections be REVERSED.

**C. CLAIMS 1, 4, 6-11, 13-15, 18, AND 20-21 ARE ALLOWABLE OVER SAND ET AL.**

The Office rejected claims 1, 4, 6-11, 13-15, 18, and 20-21 as obvious under 35 U.S.C. §103(a) over U.S. Letters Patent 6,148,322 (“Sand *et al.*”). To establish a *prima facie* case of obviousness, the prior art reference (or references when combined) must teach or suggest all the claim limitations. M.P.E.P. § 706.02(j); *In re Royka*, 490 F.2d 981, 180 U.S.P.Q. 580 (CCPA 1974). Sand *et al.* fails to meet this standard.<sup>1</sup>

**1. Sand *et al.* Fails to Disclose the Exit Routine Call Recited in Claims 1, 4, 6-7, 15, 18, and 20-21**

Claims 1 and 15 recite that the exit routine for an incomplete task is called after a preselected time period. Applicant respectfully submits that the Office has misconstrued Sand *et al.* with respect to this limitation. The Office first cites the Abstract of Sand *et al.*, which reads:

The present invention provides a processing unit with an improved ability to coordinate the execution of multiple tasks with varying priorities. Tasks to be executed are assigned both a request condition and a terminating condition, with the processing unit initiating execution of the task with the highest priority whose

---

<sup>1</sup> The United States Supreme Court today released its decision in *KSR Int'l Co. v. Teleflex Inc.*, No. 04-1350 (U.S.S.Ct. April 30, 2007), regarding the standard for obviousness under 35 U.S.C. §103. However, the decision does not address the principle on which applicant relies—namely, that obviousness requires all the limitations of the claims taught in the cited art.

request condition is satisfied. In general, *the processing unit terminates an executing task once the terminating condition of that task is satisfied*, and then initiates execution of the next highest-priority task with a satisfied request condition. However, the processing unit may abort execution of a task (other than the highest-priority task) if the request condition of a higher-priority task becomes satisfied. Moreover, the processing unit ensures the highest-priority task does not monopolize system resources by *tracking the elapsed execution time and terminating the highest-priority task if this elapsed time exceeds a predetermined maximum*, in which case the processing unit initiates execution of the next highest priority task with a satisfied request condition.

(emphasis added). Thus, the Abstract does teach termination of a task that is hung-up, but it does not teach that this occurs through call and execution of an exit routine. The Office next relies on the following passage from the detailed description:

*Processing of the higher-priority task continues until a predetermined maximum processing time elapses, at which time the task is aborted in favor of a task having a next highest priority for which the request condition is satisfied and the terminating condition is not satisfied. However, the highest-priority task can only be aborted based on the maximum time processing elapsing, and cannot be superseded by another highest-priority task (that is, another task also having the highest priority). In a variation of this embodiment, the maximum processing time may be set by the user to accommodate the particular requirements of a given system.*

(col. 2, lines 16-27; emphasis added) This passage teaches that the hung-up task is terminated by aborting it and also fails to teach or suggest that the abort occurs through call and execution of an exit routine. Finally, the Office relies on the following passage:

*Whenever the central processing unit 1 initiates execution of the high-priority task hpT, the timer module 9 begins tracking an elapsed execution time. The central processing unit 1 will abort execution of the high-priority task hpT if this elapsed time exceeds a user-selected maximum processing time, regardless of whether the terminating condition of the high-priority task hpT has been satisfied.*

(col. 4, lines 41-47; emphasis added) Again, there is no teaching that an exit routine is called when the hung-up task is aborted.

Thus, Sand *et al.* nowhere teaches or suggests that an exit routine is called when a hung-up task is aborted. Although the Office does not say so, it may be that the Office is assuming that this teaching is inherent in the teachings it cites. However, inherency and obviousness are

not synonymous since "[t]hat which may be inherent is not necessarily known. Obviousness cannot be predicated on what is unknown." *In re Newell*, 13 U.S.P.Q.2d (BNA) 1248, 1250 (Fed. Cir. 1989), quoting *In re Spormann*, 150 U.S.P.Q. (BNA) 449, 452 (C.C.P.A. 1966). "[O]ne cannot choose from the unknown." *In re Ochiai*, 37 U.S.P.Q.2d (BNA) 1127, 1131 (Fed. Cir. 1995), quoting *In re Mancy*, 182 U.S.P.Q. (BNA) 303, 306 (C.C.P.A. 1974). "Such a retrospective view of inherency is not a substitute for some teaching or suggestion supporting an obviousness rejection." *In re Rijckaert*, 28 U.S.P.Q.2d (BNA) 1955, 1957 (Fed. Cir. 1993). The principle of inherency is therefore not available to the Office in establishing *prima facie* obviousness.

The Office therefore misconstrues *Sand et al.* with respect to this limitation of claims 1 and 15, which *Sand et al.* fails to teach or suggest. Claims 4, 6-7, 18, and 20-21 incorporate this limitation by virtue of their dependence from claim 8. 35 U.S.C. §112, ¶4. *Sand et al.* therefore fails to render obvious any of claims 1, 4, 6-7, 15, 18, and 20-21 since it fails to teach or suggest each of the limitations of those claims. M.P.E.P. § 706.02(j); *In re Royka*, 490 F.2d 981, 180 U.S.P.Q. 580 (CCPA 1974).

**2. Sand et al. Fails to Disclose a "Task Picker"**  
**Recited in Claims 8-11 and 13-14**

Claim 8 expressly recites several functions of the "task picker", the accumulation of which basically means that the "task picker" controls the execution of tasks stored in the queue. For example, upon determining another task is stored in the queue, the task picker executes that task. The Office points to the teaching of the "cyclically repeated task" in *Sand et al.* as disclosing the "task picker" of Applicant's invention. However, they are not the same thing. The "cyclically repeating task" serves the purpose of insuring that every so often, a specific communication function gets executed. More importantly, it has no role in selecting which task in the queue gets selected for execution. In essence, it is a periodic interrupt that calls a communication routine.

In alleging to the contrary, the Office cites: Figure 2; col. 3, lines 45-48; col. 3, lines 56-67; and col. 4, lines 1-7. Figure 2 merely "...illustrates the relative priorities of tasks executed by a programming unit..." (col. 3, lines 1-3) The first excerpt from the specification reads:

Each task is then assigned a respective priority. Examples of these tasks for an embodiment of the present invention are illustrated in

FIG. 2, with the lowest priority task at the bottom of the list and the highest priority task at the top.

(col. 3, lines 45-48) Note that there is no mention of execution control among the task in the queue and describes in text no more than is shown in Figure 2. The second excerpt reads:

In the embodiment of FIG. 2, the task having the second-highest priority is the high-priority communications task hpKt. This task evaluates communications instructions received from the high-priority task hpT and determines whether the received communications instructions are high-priority or low-priority. If the received communications instructions are high-priority, the high-priority task hpT processes the instructions itself; otherwise, the high-priority task hpT transfers the instructions to a low-priority communications task npKt for processing. The low-priority communications task npKt is assigned the lowest priority of all of the tasks.

(col. 3, lines 56-67) In this excerpt, the various tasks in the queue pass control among themselves depending on priority—that is, no one of them controls the execution of tasks within the queue. Furthermore, none of these tasks is the “cyclically repeating task”. The third excerpt reads:

Here, the high-priority communications task hpKt is implemented as an independent task, but it could also be designed as a submodule of the high-priority task hpT.

The second-lowest priority task is the so-called cyclically-repeated task zwT. This task will be continuously executed as long as no task having a higher priority is awaiting execution.

(col. 4, lines 1-7) Here, the “cyclically repeated task” and its execution are disclosed, but there is no teaching or suggestion that it exercises any kind of control over execution of tasks in the queue.

Accordingly, the Office misconstrues Sand *et al.*, which fails to disclose a “task picker” such as that recited in claim 8. Claims 9-11 and 13-14 incorporate this limitation by virtue of their dependence from claim 8. 35 U.S.C. §112, ¶4. Sand *et al.* therefore fails to render obvious any of claims 8-11 and 13-14 since it fails to teach or suggest each of the limitations of those claims. M.P.E.P. § 706.02(j); *In re Royka*, 490 F.2d 981, 180 U.S.P.Q. 580 (CCPA 1974).

Wherefore, Applicant requests that the rejections be REVERSED.

**D. CLAIMS 2-3 AND 16-17 ARE ALLOWABLE OVER  
SAND *ET AL.* & FLETCHER**

The Office rejected claims 2-3 and 16-17 as obvious under 35 U.S.C. §103(a) over Sand *et al.* combined with U.S. Letters Patent 5,012,409 ("Fletcher"). Claims 2-3 depend from claim 1 and claims 16-17 depend from claim 15. These claims incorporate limitations of their independent claims by virtue of their dependence. 35 U.S.C. §112, ¶4. These rejections rely Sand *et al.* to each all the limitations recited in the independent claims. As is established above, Sand *et al.* fails to do so. The combination of Sand *et al.* and Fletcher consequently fails to teach or suggest each of the limitations of those claims. Accordingly, Sand *et al.* in combination with Fletcher fails to render obvious any of claims 2-3 and 16-17. M.P.E.P. § 706.02(j); *In re Royka*, 490 F.2d 981, 180 U.S.P.Q. 580 (CCPA 1974). Wherefore, Applicant requests that the rejections be REVERSED.

**VIII. CLAIMS APPENDIX**

The claims that are the subject of the present appeal – claims 1-4, 6-11, 13-18 and 20-28 – are set forth in the attached "Claims Appendix."

**IX. EVIDENCE APPENDIX**

There is no separate Evidence Appendix for this appeal.

**X. RELATED PROCEEDINGS APPENDIX**

There is no Related Proceedings Appendix for this appeal.

**XI. CONCLUSION**


Applicants therefore respectfully submit that the claims are allowable over the art of record. Accordingly, Applicants request that the rejections be overturned and the claims allowed to issue.

Please date stamp and return the enclosed postcard to evidence receipt of this document.

Respectfully submitted,

Date: April 30, 2007

WILLIAMS, MORGAN & AMERSON  
10333 Richmond Dr., Suite 1100  
Houston, Texas 77042  
(713) 934-4053 ph

  
/Jeffrey A. Pyle/  
Jeffrey A. Pyle  
Reg. No. 34,904  
Attorney for Applicants

RECEIVED  
CENTRAL FAX CENTER

APR 30 2007

APPENDIX  
(Claims in Issue)

1. A computer implemented method, comprising:  
storing one or more tasks in a queue, wherein each task has an associated exit routine;  
determining at least one task to process based on a priority scheme;  
processing the at least one task; and  
calling the exit routine based on determining that the task has not completed processing  
within a preselected period of time.
2. The computer implemented method of claim 1, wherein storing the one or more task in  
the queue comprises storing at least one task in the queue at every preselected time interval.
3. The computer implemented method of claim 1, further comprising generating an  
interrupt, and wherein storing the one or more tasks in the queue comprises storing the one or  
more tasks in the queue in response to detecting the interrupt.
4. The computer implemented method of claim 1, wherein determining at least one task to  
process based on the priority scheme comprises determining the at least one task based on a first-  
in, first-out priority scheme.
6. The computer implemented method of claim 1, wherein calling the exit routine comprises  
terminating the task currently processing and returning control to a task picker in the queue.
7. The computer implemented method of claim 1, wherein processing the at least one task  
comprises executing the task and programming a timer to generate an interrupt after a  
preselected time, wherein the preselected time corresponds to the amount of time required for the  
task to complete executing.
8. A computing apparatus, comprising:  
a queue having a task picker stored therein, the task picker being configured to:  
determine if at least one task other than the task picker is stored in the queue;

transfer control to the at least one task other than the task picker based on determining that the at least one task other than the task picker is stored in the queue so that the at least one task other than the task picker can execute; and  
execute in response to the at least one task other than the task picker completing execution and continue executing until a preselected event occurs.

9. The apparatus of claim 8, wherein the preselected event comprises detection of an interrupt.
10. The apparatus of claim 8, wherein the preselected event comprises detection of another task being present in the queue.
11. The apparatus of claim 8, wherein each task stored in the queue comprises an exit routine to terminate that task.
13. The apparatus of claim 8, wherein the task picker determines that more than one task is stored in the queue and wherein the task picker selects a task to execute from the one or more tasks based on a priority scheme.
14. The apparatus of claim 13, wherein the priority scheme is a first-in, first-out scheme.
15. An article comprising one or more machine-readable storage media containing instructions that when executed enable a processor to:
  - store one or more tasks in a storage space, wherein each task has an associated exit routine;
  - determine at least one task to process based on a priority scheme;
  - process the at least one task; and
  - call the exit routine based on determining that the task cannot be processed to completion.



16. The article of claim 15, wherein the instructions when executed enable the processor to store the one or more tasks in the storage space comprises storing at least one task in the storage space at every preselected time interval.

17. The article of claim 15, wherein the instructions when executed enable the processor to generate an interrupt and store the one or more tasks in the storage space in response to detecting the interrupt.

18. The article of claim 15, wherein the instructions when executed enable the processor to determine the at least one task based on a first-in, first-out priority scheme.

20. The article of claim 15, wherein the instructions when executed enable the processor to terminate the task currently processing and return control to a task picker in the storage space.

21. The article of claim 15, wherein the instructions when executed enable the processor to execute the task and to program a time to generate an interrupt at a preselected time, wherein the preselected time is greater than the time required for the task to complete executing.

22. An apparatus, comprising:  
a queue having a task picker stored therein, the queue adapted to store one or more tasks,  
and the task picker being configured to:  
select a task from the queue to execute based on a priority scheme;  
transfer control to the task so that the task can execute;  
a failure recovery timer to generate an interrupt at preselected time intervals,  
wherein each preselected time interval is greater than the time it takes for  
each of the tasks stored in the queue to execute; and  
a controller adapted to:  
determine if the task completes execution within the preselected time interval;  
terminate the task in response to determining that the task failed to complete  
within the preselected time interval; and  
execute the task picker in response to terminating the task.

23. The apparatus of claim 22, wherein the priority scheme is based on a first-in, first-out scheme.
24. The apparatus of claim 22, wherein each task has an associated exit routine and wherein the controller terminates the task by calling the exit routine.
25. The apparatus of claim 22, wherein the controller resets the failure recovery timer before executing the task.
26. The apparatus of claim 22, wherein the controller determines if the task completes execution within the preselected time interval comprises:
- detecting a first failure recovery interrupt;
  - causing an interrupt service routine to determine a task ID associated with a task executing at the time of the first failure recovery interrupt;
  - logging the determined task ID;
  - detecting a second failure recovery interrupt;
  - determining a task ID associated with a task executing at the time of the second failure recovery interrupt; and
  - terminating the task executing at the time of the second failure recovery interrupt in response to determining that the two task IDs are the same.
27. The apparatus of claim 22, further comprising a repetitive timer for generating interrupts on a periodic basis, wherein the controller posts a task in the queue in response to detecting an interrupt generated by the repetitive timer.
28. The apparatus of claim 22, wherein the controller resets the failure recovery timer before executing the task picker.